(2)

Distributed Systems: Interconnection and Fault Tolerance Studies
Final Report

Ashok Agrawala and Satish Tripathi
Systems Design and Analysis Group
Department of Computer Science
University of Maryland
College Park, MD 20742

Submitted to:

Army Strategic Defense Command
Huntsville, AL
Project Number DASG60-87-C-0066

1 Introduction

The goal of this project was to study the primary design and implementation issues in distributed implementation of hard real-time systems. We organized the effort under a project named *MARUTI* and defined the goal as the creation of an environment for the development and deployment of applications with hard real-time, fault tolerance, and security requirements, as are often found in the embedded systems[12, 13]. Good examples of such embedded systems are found in signal processing and avionics applications. Such applications must be able to execute on a distributed, heterogeneous hardware base. During the past three years we have created a framework for such an environment and have demonstrated the feasiblty of the design through initial implementations of the prototype components of the *MARUTI* Environment. In this proposal, we outline the research effort we propose to undertake over the next three years.

The design of the *MARUTI* Environment is motivated by the requirements of the next generation of applications. In the rest of this section we present some details of these requirements.

2 Project Accomplishments

In order to address the problems associated with the design and implementation of an advanced, hard real-time operating system, a number of new techniques have to be developed. Our approach has been to take a comprehensive view of the problems and address them at the theoretical level when necessary. At the same time we integrate such developments in implementations, and derive new theoretical challenges from our experiences in implementing the system.

In this section we present the results to date in theoretical work as well as the current status of the implementation of the *MARUTI* environment.

2.1 Theoretical Developments

The development of a comprehensive framework which addresses the requirements for hard real-time, fault tolerance, and distributed heterogeneous operation poses many new theoretical challenges. During the past three years, we have been addressing several such problems. In the following we discuss some of our achievements which are contributions to the state of the art in their own right. Clearly they have had a major impact on the design and implementations we have undertaken.

The primary paradigm in the design of the *MARUTI* environment is that of time-driven computations. Development of such an environment required a re-examination of the basic assumptions and approaches, as the generalization of current practices were not applicable. A comprehensive description of our approach has been presented in the book[2].

2.2 Resource Allocation

Clearly the resource management problem is at the heart of a successful implementation of a real-time operating system in a distributed environment. Our studies of the issues involved resulted in our separating the resource management problem into two phases, resource allocation and scheduling. In our design, an allocator decides where tasks and subtasks are to execute. The actual local scheduling of a resource is carried out in the second phase [2].

In conjunction with the allocation of resources, we have developed the concept of resource verification, which is carried out to ascertain that the scheduling constraints will be met. In this way the allocation process carrie: out its primary function of assigning tasks to various nodes, taking into consideration the timing constraints. We have studied policy issues related to the resource allocation[10].

2.3 Real-Time Scheduling

The system maintains a *calendar* for each resource. If a task's request for a resource can be satisfied, a reservation is made in the calendar. At run time, resources are allocated to tasks according to the reservations in the calendar.

In[14], we present a new technique for scheduling: *decomposition scheduling*. All the requests are decomposed into a sequence of subsets. Each request is in one and only one subset. The requests in an earlier subset of the sequence are scheduled before the requests in a later subset. Tasks within each subset are scheduled using an approach called *super-sequence scheduling*[16, 15].

In determining the task schedules and constructing the subsets, we take into account the relationships which exist among the time constraints of tasks. We have defined *leading* and *strongly-leading* relations, and carry out the decomposition based on these relations.

The results to date indicate that this scheduling approach guarantees the generation of a feasible schedule if one exists[3]. At the same time it has relatively modest computation requirements.

2.4 Fault Tolerance

The *MARUTI* system has been built as a fault tolerant, distributed system. To achieve the fault tolerance objective, processors in the system are divided into partitions. A real-time task can be executed in different partitions of processors at the same time. The allocation of tasks to partitions is dynamically determined according to system parameters, and according to how many faults the application must tolerate. The resource allocators in different sites exchange replica information to ensure correct message delivery to all replicas. A theoretical model of this scheme has been presented in[2].

The fault tolerance scheme follows a *fork-join* paradigm. A message-sending task sends its output message to all the replicated message-receiving tasks. The fork part of the task takes care of the multiple message sending, and is transparent to the user. Each receiving task has a user-transparent join part, which selects the correct message, based on time and syntax, and gives it directly to the message-receiving task. We have shown that this paradigm is applicable to handling user-definable resiliency requirements on an application by application basis[5]. In addition it gives flexibility in that a different degree of resiliency may be specified for different parts of a computation graph. The approach permits the construction of a resilient computation graph, which is capable of restoring the degree of resiliency after a transient failure[5, 6].

2.5 Programming Language Support

The time-driven approach requires the scheduler to know the resource requirements, time constraints, and execution time of each application. Communication, precedence and synchronization among processes affect the time constraints of applications, and m'ist be taken into account while scheduling. Since these constraints and requirements are application-specific, they need to be derived from application programs. Therefore, the programming language has to provide the programmer with features to express them.

MPL (*MARUTI* Programming Language)[8, 7] is based on an object-oriented paradigm[9]. MPL objects communicate with each other using both one-way method invocations or remote procedure calls. It provides exception handling including timing errors. It provides features to express time constraints on invocations and precedence relations among them. This information is used for pre-scheduling. MPL provides separate type hierarchy and inheritance hierarchy. It is possible to have multiple implementations for a given object specification. The MPL objects provide intra-object as well as inter-object concurrency. It is also possible to express that certain actions have to occur in parallel or simultaneously. The synchronization mechanism is also designed to facilitate pre-scheduling. The language supports fault tolerance using strong typing, using exception handling, and creating object groups. An object group is a mechanism to address, communicate, and control a number of cooperating objects.

Apart from translation, the MPL compiler extracts the temporal and synchronization constraints of objects. These are later used by the scheduler to create a calendar.

2.6 Implementation

MARUTI is built as a modular system and it allows the design, analysis and verification of properties of user applications executable in the system. It is also designed to be deterministic and predictable. The implementation is carried out according to these design goals[4, 11].

MARUTI has been demonstrated as a distributed, real-time, fault tolerant system in a heterogeneous environment. In *MARUTI* real-time tasks and system services are distributed among a set of processors. As a result, it is not necessary to keep a copy of each service in each processor. The local allocator dynamically decides to invoke a service on any machine. Furthermore when a local service cannot meet all the local requests, that is, the service cannot meet its deadline for every request from the local tasks, it invokes the same service on a remote machine. The remote service coordination is carried out by the allocators in local and remote processors, through a process of negotiation.

The *MARUTI* system is designed as a fault tolerant system. We divide processors into several partitions, such that no fault propagation can take place from one partition to another. One copy of a real-time application is only run within one partition. For the fault tolerant purpose, multiple copies of an application may run at the same time in different partitions. We have implemented the fork-join mechanism discussed above to provide the fault tolerance function.

The *MARUTI* system is designed as a heterogeneous system. It is implemented on different machines, including Sun-3, SparcStations, and DECStations. Since different machines have different formats of number storage, we are developing various tools which can translate between different formats when communication is needed between different machines.

The current implementation of *MARUTI* runs on top of the UNIX[1] system. While UNIX is not the most hospitable host to implement a time-driven system such as *MARUTI*, it offers a very effective system development environment. Further the availability of the UNIX system on many platforms makes *MARUTI* portable. Experience gained in building *MARUTI* on top of UNIX have been documented in[11].

2.7 Tools

The *MARUTI* environment contains a set of tools which have been developed to support the applications during all phases of their life cycle. The following tools are available in the system at present:

- *Precompiler.* In order to run the real-time language we have developed, we have built a precompiler which can translate code from our language into C code.

- *Joint Editor.* Currently, part of the information contained in the joints, such as execution times of SAPs and their temporal relations with other SAPs, has to be created and updated manually. In order to simplify this task, an interactive joint editor is provided. In the next version, most of this information will be created by the execution time analyzer and the precompiler.

- *Scheduling Tool.* During the application development it is necessary for the programmer to get some idea about the ability of the program to execute with the necessary time constraints. In order to support such analysis the scheduling tool permits a user to study the schedulability of a set of tasks on a set of system resources. The task characteristics are specied as a computation graph with the resource requirements for each node of the graph given explicitly. The tool then examines the feasibility of scheduling the tasks with the time constrains and provides an analysis of the resource bottlenecks and the application program bottlenecks. An initial version of this tool is operational.

- *Calendar Display.* This tool presents a dynamic display of the current schedule of tasks to be executed. It also supports a step by step controlled execution of tasks. This tool and its displays have been very useful in debugging the demo applications in that its use has become an integral part of *MARUTI* demos.

  Since time is one of the most important factors in the system, we have developed a tool which can stop the *MARUTI* system timer. The tool can also be used to run tasks in stepwise fashion. That is, tasks can be run one by one when a user selects a button on the screen.

The *MARUTI* system has provided us a platform on which to try different ideas and to implement different tools. We plan to build more tools, such as an automatic task execution time analyzer. This would be run at compile time, and would use the syntax of the code to determine the execution time of real-time tasks.

References

[1] S. J. Leffler, M. K. McKusick, M. J. Karels, and J. S. Quaterman. *The Design and Implementation of the 4.3BSD* UNIX *Operating System.* Addison-Wesley, 1989.

[2] Shem-Tov Levi and Ashok K. Agrawala. *Real Time System Design.* McGraw-Hill, New York N.Y, 1990.

[3] Manas C. Saksena and Ashok K. Agrawala. Temporal analysis and its application in non-preemptive scheduling. Technical report, U of MD, July 1991.

[4] Daniel Mossé, Ólafur Gudmundsson, and Ashok K. Agrawala. The *MARUTI* system and its implementation. Technical Report CS TR 2694, U of MD, June 1991.

[5] Daniel Mossé and Ashok K. Agrawala. Resilient computation graphs for fault tolerance in real-time environments. Technical Report CS-TR-2613, UMIACS-TR-91-29, U of MD, February 1991.

[6] Noura F. Zoubeir. Fault-tolerance implementation for maruti, a real-time distributed operating system. Technical report, U of MD, July 1991.

[7] Vivek Nirkhe and William Pugh. A Partial Evaluator for the Maruti Real-Time System. In *Real-Time Systems Symposium*, 1991. Submitted for Publication.

[8] Vivek Nirkhe, Satish Tripathi, and Ashok Agrawala. Language Support for the Maruti Real-Time System. In *Real-Time Systems Symposium*, December 1990.

[9] Vivek Nirkhe and Satish K. Tripathi. Language Support for Maruti Real-Time System. Technical Report CS-TR-2481, University of Maryland, College Park MD 20742, 1990.

[10] Ólafur Gudmundsson, Keng-Tai Ko, Yiheng Shi, and Ashok K. Agrawala. On resource management in hard real-time distributed operating systems. Technical report, University of Maryland, College Park MD 20742, January 1991.

[11] Ólafur Gudmundsson, Daniel Mossé, Ashok K. Agrawala, and Satish K. Tripathi. *MARUTI* a hard real-time operating system. In *Second IEEE Workshop on Experimental Distributed Systems*, pages 29–34. IEEE, 1990.

[12] Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi. Maruti an environment for hard real-time applications. Technical Report CS-TR-2328, Department of Computer Science, University of Maryland, College Park, Maryland, Nov. 1989.

[13] J. A. Stankovic. Misconceptions about real-time computing: A serious problem for next-generation systems. *IEEE Computer*, 21(10):10–19, Oct. 1988.

[14] Xiaoping George Yuan and Ashok K. Agrawala. A decompostion approach to nonpreemptive real-time scheduling. Technical Report CS-TR-2345, umd, Nov. 1989.

[15] X. Yuan and A. K. Agrawala. Decomposition with the strongly-leading relation for hard real-time scheduling. Technical Report CS-TR-2346, Dept. of Computer Science, Univ. of Maryland, Coll. Pk., MD 20742, Nov. 1989.

[16] X. Yuan and A. K. Agrawala. Scheduling real-time task in single schedule subsets. Technical Report CS-TR-2347, Dept. of Computer Science, Univ. of Maryland, Coll. Pk., MD 20742, Nov. 1989.

A Publications of the *MARUTI* Project

The *MARUTI* project has already lead to interesting theoretical and practical results. These are documented in the following books, articles, and technical reports.

- Vivek Nirkhe and William Pugh, " Partial Evaluation of high level imperative progammning languages with applications in Hard Real-Time systems", ACM conference Principles of Programming Languages, January, 1992.
- Daniel Mossé, Ólafur Gudmundsson, and Ashok K. Agrawala, "Prototyping Real Time Operating Systems", First International Workshop on Rapid System Prototyping, IEEE-CS, 1991.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala and Satish K. Tripathi, " MARUTI: A Platform for Hard Real-Time Applications", Mission Critical Operating Systems ,IOS Press, 1991 A.K. Agrawala, K. Gordan and P. Hwang (eds.).
- Vivek Nirkhe and William Pugh " A Partial Evaluator for the MARUTI Hard Real-Time System", 12th IEEE Real-Time Systems Symposium San Antonio, TX, December, 1991.
- S. Rangarajan and Satish K. Tripathi, "Efficient Synchronization of Clocks in a Distributed System", 12th IEEE Real-Time Systems Symposium San Antonio, TX, December, 1991.
- Satish K. Tripathi and Vivek Nirkhe, " Synchronization in Hard Real-Time Systems (Position Paper)", Operating Systems of the 90's and Beyond, Dagstuhl Castle, Germany, July 1991.
- Noura F. Zoubeir, " Fault-Tolerance Implementation for MARUTI, a Real-Time Distributed Operating System",, CS-TR-2728, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July, 1991.
- Andrew P. Balinsky, " Adding Heterogeneous Communication Support to MARUTI: a Real-Time Operating System", CS-TR-2710, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Manas C. Saksena and Ashok K. Agrawala, " Temporal Analysis and its Application in Non-Preemptive Scheduling", CS-TR-2698, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Andrew P. Balinsky, Ólafur Gudmundsson and Ashok K. Agrawala, " Real-Time Heterogeneous Communication Support in MARUTI", CS-TR-2697, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Daniel Mossé and Ólafur Gudmundsson and Ashok K. Agrawala, " The MARUTI System and its Implementation", CS-TR-2694, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Partho Pratim Mishra, Ólafur Gudmundsson and Ashok K. Agrawala, "Exmon: A tool for resource montioring of programs", CS-TR-2688, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.

- Vivek Nirkhe and William Pugh " Application of Partial Evaluation to Hard Real-Time Programming", Eighth IEEE Workshop on Real-Time Operating Systems and Software and 17th IFAC/IFIP Workshop on Real-Time Programming, Pergamon Press, May, 1991.
- Daniel Mossé and Ashok K. Agrawala. " Resilient Computation Graphs for Fault Tolerance in Real-Time Environments", CS-TR-2613, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February, 1991.
- Ólafur Gudmundsson, Keng-Tai Ko, Yiheng Shi and Ashok K. Agrawala, " On Resource Management in Hard Real-Time Distributed Operating Systems", CS-TR-2582, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, January, 1991.
- Sarit Mukherjee, Satish K. Tripathi and D. Ghosal, "Performance Analysis of a Multiclass Priority-based Slotted Ring LAN", CS-TR-2581, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, January, 1991.
- Satish K. Tripathi and Vivek Nirkhe, " Synchronization in Hard Real-Time Systems", Frontiers in Computing Systems Research - Essays on Emerging Technologies, Architectures and Theories, Plenum Press, 1990, S. K. Tewksbury (Ed).
- Vivek Nirkhe, Satish K. Tripathi and Ashok K. Agrawala, "Language Support for the Maruti Real-Time System", 11th IEEE Real-Time Systems Symposium, Orlando, FL, pages="257-266", December, 1990.
- Ashok K. Agrawala and J. Hendler " Mission Critical Planning: AI on the MARUTI Real-Time Operating System", Proceedings of DARPA Planning Workshop, November, 1990.
- Ólafur Gudmundsson, Daniel Mossé, Ashok K. Agrawala, and Satish K. Tripathi, "MARUTI: A Hard Real-Time Operating System", IEEE Workshop on Experimental Distributed Systems, Huntsville, AL, October 1990.
- Ólafur Gudmundsson, Dheeraj Sanghi, Ashok K. Agrawala, and Thareja A., "Invisible Resource Usage in UNIX", CS-TR-2509, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July 1990.
- Vivek Nirkhe, Satish K. Tripathi and Ashok K. Agrawala, "Language Support for Maruti Real-Time System", CS-TR-2481, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, May 1990.
- Xiaoping George Yuan and Ashok K. Agrawala, "Evaluation of a Decomposition Approach for Real-Time Scheduling Using a Stochastic Model", CS-TR-2462, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1990.
- Shem-Tov Levi and Ashok K. Agrawala, "Real-Time System Design", McGraw Hill Publishing Co, New York, 1990.
- Xiaoping George Yuan and Ashok K. Agrawala, "A Decomposition Approach to Nonpreemptive Scheduling in Hard Real-Time Systems", 11th IEEE Real-Time Systems Symposium, Orlando, FL, IEEE, December 1989.

- Huang Y. and Satish K. Tripathi, "Resource Allocation for Fault Tolerant Systems Using External Backups.", CS-TR-2343, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ashok K. Agrawala, Ólafur Gudmundsson, and Daniel Mossé, "Mission Critical Operating Systems Requirements and the *MARUTI* Project", CS-TR-2342, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ashok K. Agrawala and C. Kim, "Time Estimates and Clock Synchronization in Distributed Systems", CS-TR-2348, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November, 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Scheduling Real-Time Tasks in Single Schedule Subsets", CS-TR-2347, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Decomposition with a Strongly-Leading Relation for Hard Real-Time Scheduling", CS-TR-2346, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "A Decomposition Approach to Nonpreemptive Real-Time Scheduling", CS-TR-2345 Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Satish K. Tripathi and Y. Huang, "Resource Allocation for Fault Tolerant Systems Using External Backups", CS-TR-2343, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi , " MARUTI: A Platform for Hard Real-Time Applications", CS-TR-2342, Computer Science Dept, University of Maryland, College Park, Maryland, October 1989.
- Vivek Nirkhe and Satish K. Tripathi, "Synchronization in Hard Real-Time Systems.", CS-TR-2337, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, October 1989.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi, "*MARUTI* an Environment for Hard Real-Time Applications", CS-TR-2328, Department of Computer Science, University of Maryland, College Park, Maryland, October 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Real-Time scheduling with Both Preemption and Nonpreemption Requirements", Proceedings of the Fifteenth Symposium on Microprocessing and Microprogramming, Koln, F. R. Germany, September, 1989.
- Shem-Tov Levi, Satish K. Tripathi, Scott D. Carson, and Ashok K. Agrawala, "The MARUTI Hard Real-Time Operating System", ACM Operating System Review, June 1989, Vol 23, No 3.

- Xiaoping George Yuan and Ashok K. Agrawala, "Real-Time Scheduling with Both Preemption and Nonpreemption Requirements", CS-TR-2248, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1989.
- Yuan S.M. and Ashok K. Agrawala, "An Efficient Communication Structure for Decentralized Algorithms with Fault Tolerance", CS-TR-2206, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1989.
- Sam H. Noh and Ashok K. Agrawala, "Process Timing in UNIX", CS-TR-2205, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1989.
- Ashok K. Agrawala and Shem-Tov Levi, " Distributed Real-Time Operating Systems", McGraw-Hill, Inc. New York, NY 1989.
- Nehmer J., "A Structuring Framework for Distributed Operating Systems", CS-TR-2079, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July 1988.
- Shem-Tov Levi, Daniel Mossé, and Ashok K. Agrawala, "Allocation of Real-Time Computations under Fault Tolerance Constraints", CS-TR-2018, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, May 1988.
- Shem-Tov Levi, Ashok K. Agrawala, and Satish K. Tripathi, "Introducing the MARUTI Hard Real-Time Operating System", CS-TR-2010, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1988.
- J. Nehmer, "An Object Architecture for Hard Real-Time Systems", CS-TR-2003, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, March 1988.
- Chintamaneni P., Xiaoping George Yuan, Satish K. Tripathi , and Ashok K. Agrawala, "Scheduling Tasks in a Real-Time System", CS-TR-1991, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1988.
- Shem-Tov Levi, Daniel Mossé, and Ashok K. Agrawala, "Resource Allocation under Fault-Tolerance Constraints", Proceedings of the IEEE Real-Time Systems Symposium, Huntsville, AL, 1988
- Shem-Tov Levi, "A Methodology for Designing Distributed, Fault-Tolerant, Reactive, Real-Time Operating Systems", PhD. Dissertation University of Maryland, College Park, MD. 1988.
- Xiaoping George Yuan, Satish K. Tripathi, and Ashok K. Agrawala, "Scheduling in Real-Time Distributed Systems-A Review", CS-TR-1955, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, December 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Temporal Relations and Structures in Real-Time Operating Systems", CS-TR-1954, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, December 1987.

- Shieh Y., Satish K. Tripathi, Chintamaneni P. and Pankaj Jalote, "On Fault Tolerance in Manufacturing Systems",, CS-TR-1939, Technical Report, Department of Computer Science, University of Maryland. College Park, Maryland, October 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Objects Architecture: A Comprehensive Design Approach for Real-Time, Distributed, Fault-Tolerant, Reactive Operating Systems", CS-TR-1915, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, September, 1987.
- Finkel D. and Satish K. Tripathi, "An Analysis of a Buddy System for Fault Tolerance", CS-TR-1924, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, August 1987.
- Shem-Tov Levi and Ashok K. Agrawala, " On Real-Time Systems Using Local Area Networks", CS-TR-1892, Technical Report, Department of Computer Science, UMIACS-TR-87-35, Technical Report, Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland, July 1987.
- Ashok K. Agrawala and Shem-Tov Levi, "Objects Architecture for Real-Time, Distributed, Fault Tolerant Operating Systems", IEEE Workshop on Real-Time Operating Systems, Cambridge, MA, July 1987, pp. 142-148.
- Shem-Tov Levi and Ashok K. Agrawala, " On Real-Time Operating Systems", CS-TR-1838, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Real-Time Programs: Design Implementation and Validation-A Survey", CS-TR-1837, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1987.

UNIVERSITY OF MARYLAND AT COLLEGE PARK

DEPARTMENT OF COMPUTER SCIENCE

Ashok K. Agrawala
Phone:(301) 405-2665
Fax:(301) 405-6707
agrawala@cs.umd.edu

October 24, 1991

Dr. John Johnson
U.S. Army Strategic Defense Command
CSSD-H-V
P.O. Box 1500
Huntsville, AL 35807-3801

Reference- Contract DASG60-87-C-0066

Dear Dr. Johnson:

Enclosed please find the copies of the Final Report for the project *Distributed Systems: Interconnections and Fault Tolerance.* I am forwarding copies of the report to the distribution list as required in the contract. If you need any additional information please do not hesitate to contact me.

Sincerely yours,

Ashok K. Agrawala
Professor of
Computer Science

Copy to:
Office of Research Administration, UMCP
U.S. Army Strategic Defense Cmd., CSSD-H-V, CSSD-H-CRS, CSSD-BM-PP
Director, Defense Research Projects, 1400 Wilson Blvd, Arlington, VA
Institute for Defense Analysis, Alexandria, VA
ONR Resident Representative

Distributed Systems: Interconnection and Fault Tolerance Studies
Final Report

Ashok Agrawala and Satish Tripathi
Systems Design and Analysis Group
Department of Computer Science
University of Maryland
College Park, MD 20742

Submitted to:

Army Strategic Defense Command
Huntsville, AL
Project Number DASG60-87-C-0066

1 Introduction

The goal of this project was to study the primary design and implementation issues in distributed implementation of hard real-time systems. We organized the effort under a project named *MARUTI* and defined the goal as the creation of an environment for the development and deployment of applications with hard real-time, fault tolerance, and security requirements, as are often found in the embedded systems[12, 13]. Good examples of such embedded systems are found in signal processing and avionics applications. Such applications must be able to execute on a distributed, heterogeneous hardware base. During the past three years we have created a framework for such an environment and have demonstrated the feasiblty of the design through initial implementations of the prototype components of the *MARUTI* Environment. In this proposal, we outline the research effort we propose to undertake over the next three years.

The design of the *MARUTI* Environment is motivated by the requirements of the next generation of applications. In the rest of this section we present some details of these requirements.

2 Project Accomplishments

In order to address the problems associated with the design and implementation of an advanced, hard real-time operating system, a number of new techniques have to be developed. Our approach has been to take a comprehensive view of the problems and address them at the theoretical level when necessary. At the same time we integrate such developments in implementations, and derive new theoretical challenges from our experiences in implementing the system.

In this section we present the results to date in theoretical work as well as the current status of the implementation of the *MARUTI* environment.

2.1 Theoretical Developments

The development of a comprehensive framework which addresses the requirements for hard real-time, fault tolerance, and distributed heterogeneous operation poses many new theoretical challenges. During the past three years, we have been addressing several such problems. In the following we discuss some of our achievements which are contributions to the state of the art in their own right. Clearly they have had a major impact on the design and implementations we have undertaken.

The primary paradigm in the design of the *MARUTI* environment is that of time-driven computations. Development of such an environment required a re-examination of the basic assumptions and approaches, as the generalization of current practices were not applicable. A comprehensive description of our approach has been presented in the book[2].

2.2 Resource Allocation

Clearly the resource management problem is at the heart of a successful implementation of a real-time operating system in a distributed environment. Our studies of the issues involved resulted in our separating the resource management problem into two phases, resource allocation and scheduling. In our design, an allocator decides where tasks and subtasks are to execute. The actual local scheduling of a resource is ca-ried out in the second phase [2].

In conjunction with the allocation of resources, we have developed the concept of resource verification, which is carried out to ascertain that the scheduling constraints will be met. In this way the allocation process carries out its primary function of assigning tasks to various nodes, taking into consideration the timing constraints. We have studied policy issues related to the resource allocation[10].

2.3 Real-Time Scheduling

The system maintains a *calendar* for each resource. If a task's request for a resource can be satisfied, a reservation is made in the calendar. At run time, resources are allocated to tasks according to the reservations in the calendar.

In[14], we present a new technique for scheduling: *decomposition scheduling.* All the requests are decomposed into a sequence of subsets. Each request is in one and only one subset. The requests in an earlier subset of the sequence are scheduled before the requests in a later subset. Tasks within each subset are scheduled using an approach called *super-sequence scheduling*[16, 15].

In determining the task schedules and constructing the subsets, we take into account the relationships which exist among the time constraints of tasks. We have defined *leading* and *strongly-leading* relations, and carry out the decomposition based on these relations.

The results to date indicate that this scheduling approach guarantees the generation of a feasible schedule if one exists[3]. At the same time it has relatively modest computation requirements.

2.4 Fault Tolerance

The *MARUTI* system has been built as a fault tolerant, distributed system. To achieve the fault tolerance objective, processors in the system are divided into partitions. A real-time task can be executed in different partitions of processors at the same time. The allocation of tasks to partitions is dynamically determined according to system parameters, and according to how many faults the application must tolerate. The resource allocators in different sites exchange replica information to ensure correct message delivery to all replicas. A theoretical model of this scheme has been presented in[2].

The fault tolerance scheme follows a *fork-join* paradigm. A message-sending task sends its output message to all the replicated message-receiving tasks. The fork part of the task takes care of the multiple message sending, and is transparent to the user. Each receiving task has a user-transparent join part, which selects the correct message, based on time and syntax, and gives it directly to the message-receiving task. We have shown that this paradigm is applicable to handling user-definable resiliency requirements on an application by application basis[5]. In addition it gives flexibility in that a different degree of resiliency may be specified for different parts of a computation graph. The approach permits the construction of a resilient computation graph, which is capable of restoring the degree of resiliency after a transient failure[5, 6].

2.5 Programming Language Support

The time-driven approach requires the scheduler to know the resource requirements, time constraints, and execution time of each application. Communication, precedence and synchronization among processes affect the time constraints of applications, and must be taken into account while scheduling. Since these constraints and requirements are application-specific, they need to be derived from application programs. Therefore, the programming language has to provide the programmer with features to express them.

MPL (*MARUTI* Programming Language)[8, 7] is based on an object-oriented paradigm[9]. MPL objects communicate with each other using both one-way method invocations or remote procedure calls. It provides exception handling including timing errors. It provides features to express time constraints on invocations and precedence relations among them. This information is used for pre-scheduling. MPL provides separate type hierarchy and inheritance hierarchy. It is possible to have multiple implementations for a given object specification. The MPL objects provide intra-object as well as inter-object concurrency. It is also possible to express that certain actions have to occur in parallel or simultaneously. The synchronization mechanism is also designed to facilitate pre-scheduling. The language supports fault tolerance using strong typing, using exception handling, and creating object groups. An object group is a mechanism to address, communicate, and control a number of cooperating objects.

Apart from translation, the MPL compiler extracts the temporal and synchronization constraints of objects. These are later used by the scheduler to create a calendar.

2.6 Implementation

MARUTI is built as a modular system and it allows the design, analysis and verification of properties of user applications executable in the system. It is also designed to be deterministic and predictable. The implementation is carried out according to these design goals[4, 11].

MARUTI has been demonstrated as a distributed, real-time, fault tolerant system in a heterogeneous environment. In *MARUTI* real-time tasks and system services are distributed among a set of processors. As a result, it is not necessary to keep a copy of each service in each processor. The local allocator dynamically decides to invoke a service on any machine. Furthermore when a local service cannot meet all the local requests, that is, the service cannot meet its deadline for every request from the local tasks, it invokes the same service on a remote machine. The remote service coordination is carried out by the allocators in local and remote processors, through a process of negotiation.

The *MARUTI* system is designed as a fault tolerant system. We divide processors into several partitions, such that no fault propagation can take place from one partition to another. One copy of a real-time application is only run within one partition. For the fault tolerant purpose, multiple copies of an application may run at the same time in different partitions. We have implemented the fork-join mechanism discussed above to provide the fault tolerance function.

The *MARUTI* system is designed as a heterogeneous system. It is implemented on different machines, including Sun-3, SparcStations, and DECStations. Since different machines have different formats of number storage, we are developing various tools which can translate between different formats when communication is needed between different machines.

The current implementation of *MARUTI* runs on top of the UNIX[1] system. While UNIX is not the most hospitable host to implement a time-driven system such as *MARUTI*, it offers a very effective system development environment. Further the availability of the UNIX system on many platforms makes *MARUTI* portable. Experience gained in building *MARUTI* on top of UNIX have been documented in[11].

2.7 Tools

The *MARUTI* environment contains a set of tools which have been developed to support the applications during all phases of their life cycle. The following tools are available in the system at present:

- *Precompiler.* In order to run the real-time language we have developed, we have built a precompiler which can translate code from our language into C code.
- *Joint Editor.* Currently, part of the information contained in the joints, such as execution times of SAPs and their temporal relations with other SAPs, has to be created and updated manually. In order to simplify this task, an interactive joint editor is provided. In the next version, most of this information will be created by the execution time analyzer and the precompiler.
- *Scheduling Tool.* During the application development it is necessary for the programmer to get some idea about the ability of the program to execute with the necessary time constraints. In order to support such analysis the scheduling tool permits a user to study the schedulability of a set of tasks on a set of system resources. The task characteristics are specied as a computation graph with the resource requirements for each node of the graph given explicitly. The tool then examines the feasibility of scheduling the tasks with the time constrains and provides an analysis of the resource bottlenecks and the application program bottlenecks. An initial version of this tool is operational.
- *Calendar Display.* This tool presents a dynamic display of the current schedule of tasks to be executed. It also supports a step by step controlled execution of tasks. This tool and its displays have been very useful in debugging the demo applications in that its use has become an integral part of *MARUTI* demos.

  Since time is one of the most important factors in the system, we have developed a tool which can stop the *MARUTI* system timer. The tool can also be used to run tasks in stepwise fashion. That is, tasks can be run one by one when a user selects a button on the screen.

The *MARUTI* system has provided us a platform on which to try different ideas and to implement different tools. We plan to build more tools, such as an automatic task execution time analyzer. This would be run at compile time, and would use the syntax of the code to determine the execution time of real-time tasks.

References

[1] S. J. Leffler, M. K. McKusick, M. J. Karels, and J. S. Quaterman. *The Design and Implementation of the 4.3BSD* UNIX *Operating System.* Addison-Wesley, 1989.

[2] Shem-Tov Levi and Ashok K. Agrawala. *Real Time System Design.* McGraw-Hill, New York N.Y, 1990.

[3] Manas C. Saksena and Ashok K. Agrawala. Temporal analysis and its application in non-preemptive scheduling. Technical report, U of MD, July 1991.

[4] Daniel Mossé, Ólafur Gudmundsson, and Ashok K. Agrawala. The *MARUTI* system and its implementation. Technical Report CS TR 2694, U of MD, June 1991.

[5] Daniel Mossé and Ashok K. Agrawala. Resilient computation graphs for fault tolerance in real-time environments. Technical Report CS-TR-2613, UMIACS-TR-91-29, U of MD, February 1991.

[6] Noura F. Zoubeir. Fault-tolerance implementation for maruti, a real-time distributed operating system. Technical report, U of MD, July 1991.

[7] Vivek Nirkhe and William Pugh. A Partial Evaluator for the Maruti Real-Time System. In *Real-Time Systems Symposium*, 1991. Submitted for Publication.

[8] Vivek Nirkhe, Satish Tripathi, and Ashok Agrawala. Language Support for the Maruti Real-Time System. In *Real-Time Systems Symposium*, December 1990.

[9] Vivek Nirkhe and Satish K. Tripathi. Language Support for Maruti Real-Time System. Technical Report CS-TR-2481, University of Maryland, College Park MD 20742, 1990.

[10] Ólafur Gudmundsson, Keng-Tai Ko, Yiheng Shi, and Ashok K. Agrawala. On resource management in hard real-time distributed operating systems. Technical report, University of Maryland, College Park MD 20742, January 1991.

[11] Ólafur Gudmundsson, Daniel Mossé, Ashok K. Agrawala, and Satish K. Tripathi. *MARUTI* a hard real-time operating system. In *Second IEEE Workshop on Experimental Distributed Systems*, pages 29–34. IEEE, 1990.

[12] Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi. Maruti an environment for hard real-time applications. Technical Report CS-TR-2328, Department of Computer Science, University of Maryland, College Park, Maryland, Nov. 1989.

[13] J. A. Stankovic. Misconceptions about real-time computing: A serious problem for next-generation systems. *IEEE Computer*, 21(10):10–19, Oct. 1988.

[14] Xiaoping George Yuan and Ashok K. Agrawala. A decompostion approach to nonpreemptive real-time scheduling. Technical Report CS-TR-2345, umd, Nov. 1989.

[15] X. Yuan and A. K. Agrawala. Decomposition with the strongly-leading relation for hard real-time scheduling. Technical Report CS-TR-2346, Dept. of Computer Science, Univ. of Maryland, Coll. Pk., MD 20742, Nov. 1989.

[16] X. Yuan and A. K. Agrawala. Scheduling real-time task in single schedule subsets. Technical Report CS-TR-2347, Dept. of Computer Science, Univ. of Maryland, Coll. Pk., MD 20742, Nov. 1989.

A Publications of the *MARUTI* Project

The *MARUTI* project has already lead to interesting theoretical and practical results. These are documented in the following books, articles, and technical reports.

- Vivek Nirkhe and William Pugh, " Partial Evaluation of high level imperative progammning languages with applications in Hard Real-Time systems", ACM conference Principles of Programming Languages, January, 1992.
- Daniel Mossé, Ólafur Gudmundsson, and Ashok K. Agrawala, "Prototyping Real Time Operating Systems", First International Workshop on Rapid System Prototyping, IEEE-CS, 1991.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala and Satish K. Tripathi, " MARUTI: A Platform for Hard Real-Time Applications", Mission Critical Operating Systems ,IOS Press, 1991 A.K. Agrawala, K. Gordan and P. Hwang (eds.).
- Vivek Nirkhe and William Pugh " A Partial Evaluator for the MARUTI Hard Real-Time System", 12th IEEE Real-Time Systems Symposium San Antonio, TX, December, 1991.
- S. Rangarajan and Satish K. Tripathi, "Efficient Synchronization of Clocks in a Distributed System", 12th IEEE Real-Time Systems Symposium San Antonio, TX, December, 1991.
- Satish K. Tripathi and Vivek Nirkhe, " Synchronization in Hard Real-Time Systems (Position Paper)", Operating Systems of the 90's and Beyond, Dagstuhl Castle, Germany, July 1991.
- Noura F. Zoubeir, " Fault-Tolerance Implementation for MARUTI, a Real-Time Distributed Operating System",, CS-TR-2728, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July, 1991.
- Andrew P. Balinsky, " Adding Heterogeneous Communication Support to MARUTI: a Real-Time Operating System", CS-TR-2710, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Manas C. Saksena and Ashok K. Agrawala, " Temporal Analysis and its Application in Non-Preemptive Scheduling", CS-TR-2698, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Andrew P. Balinsky, Ólafur Gudmundsson and Ashok K. Agrawala, " Real-Time Heterogeneous Communication Support in MARUTI", CS-TR-2697, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Daniel Mossé and Ólafur Gudmundsson and Ashok K. Agrawala, " The MARUTI System and its Implementation", CS-TR-2694, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.
- Partho Pratim Mishra, Ólafur Gudmundsson and Ashok K. Agrawala, "Exmon: A tool for resource montioring of programs", CS-TR-2688, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, June, 1991.

- Vivek Nirkhe and William Pugh " Application of Partial Evaluation to Hard Real-Time Programming", Eighth IEEE Workshop on Real-Time Operating Systems and Software and 17th IFAC/IFIP Workshop on Real-Time Programming, Pergamon Press, May, 1991.
- Daniel Mossé and Ashok K. Agrawala. " Resilient Computation Graphs for Fault Tolerance in Real-Time Environments", CS-TR-2613, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February, 1991.
- Ólafur Gudmundsson, Keng-Tai Ko, Yiheng Shi and Ashok K. Agrawala, " On Resource Management in Hard Real-Time Distributed Operating Systems", CS-TR-2582, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, January, 1991.
- Sarit Mukherjee, Satish K. Tripathi and D. Ghosal, "Performance Analysis of a Multiclass Priority-based Slotted Ring LAN", CS-TR-2581, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, January, 1991.
- Satish K. Tripathi and Vivek Nirkhe, " Synchronization in Hard Real-Time Systems", Frontiers in Computing Systems Research - Essays on Emerging Technologies, Architectures and Theories, Plenum Press, 1990, S. K. Tewksbury (Ed).
- Vivek Nirkhe, Satish K. Tripathi and Ashok K. Agrawala, "Language Support for the Maruti Real-Time System", 11th IEEE Real-Time Systems Symposium, Orlando, FL, pages="257-266", December, 1990.
- Ashok K. Agrawala and J. Hendler " Mission Critical Planning: AI on the MARUTI Real-Time Operating System", Proceedings of DARPA Planning Workshop, November, 1990.
- Ólafur Gudmundsson, Daniel Mossé, Ashok K. Agrawala, and Satish K. Tripathi, "MARUTI: A Hard Real-Time Operating System", IEEE Workshop on Experimental Distributed Systems, Huntsville, AL, October 1990.
- Ólafur Gudmundsson, Dheeraj Sanghi, Ashok K. Agrawala, and Thareja A., "Invisible Resource Usage in UNIX", CS-TR-2509, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July 1990.
- Vivek Nirkhe, Satish K. Tripathi and Ashok K. Agrawala, "Language Support for Maruti Real-Time System", CS-TR-2481, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, May 1990.
- Xiaoping George Yuan and Ashok K. Agrawala, "Evaluation of a Decomposition Approach for Real-Time Scheduling Using a Stochastic Model", CS-TR-2462, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1990.
- Shem-Tov Levi and Ashok K. Agrawala, "Real-Time System Design", McGraw Hill Publishing Co, New York, 1990.
- Xiaoping George Yuan and Ashok K. Agrawala, "A Decomposition Approach to Nonpreemptive Scheduling in Hard Real-Time Systems", 11th IEEE Real-Time Systems Symposium, Orlando, FL, IEEE, December 1989.

- Huang Y. and Satish K. Tripathi, "Resource Allocation for Fault Tolerant Systems Using External Backups.", CS-TR-2343, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ashok K. Agrawala, Ólafur Gudmundsson, and Daniel Mossé, "Mission Critical Operating Systems Requirements and the *MARUTI* Project", CS-TR-2342, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ashok K. Agrawala and C. Kim, "Time Estimates and Clock Synchronization in Distributed Systems", CS-TR-2348, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November, 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Scheduling Real-Time Tasks in Single Schedule Subsets", CS-TR-2347, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Decomposition with a Strongly-Leading Relation for Hard Real-Time Scheduling", CS-TR-2346, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "A Decomposition Approach to Nonpreemptive Real-Time Scheduling", CS-TR-2345 Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Satish K. Tripathi and Y. Huang, "Resource Allocation for Fault Tolerant Systems Using External Backups", CS-TR-2343, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, November 1989.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi , " MARUTI: A Platform for Hard Real-Time Applications", CS-TR-2342, Computer Science Dept, University of Maryland, College Park, Maryland, October 1989.
- Vivek Nirkhe and Satish K. Tripathi, "Synchronization in Hard Real-Time Systems.", CS-TR-2337, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, October 1989.
- Ólafur Gudmundsson, Daniel Mossé, Keng-Tai Ko, Ashok K. Agrawala, and Satish K. Tripathi, "*MARUTI* an Environment for Hard Real-Time Applications", CS-TR-2328, Department of Computer Science, University of Maryland, College Park, Maryland, October 1989.
- Xiaoping George Yuan and Ashok K. Agrawala, "Real-Time scheduling with Both Preemption and Nonpreemption Requirements", Proceedings of the Fifteenth Symposium on Microprocessing and Microprogramming, Koln, F. R. Germany, September, 1989.
- Shem-Tov Levi, Satish K. Tripathi, Scott D. Carson, and Ashok K. Agrawala, "The MARUTI Hard Real-Time Operating System", ACM Operating System Review, June 1989, Vol 23, No 3.

- Xiaoping George Yuan and Ashok K. Agrawala, "Real-Time Scheduling with Both Preemption and Nonpreemption Requirements", CS-TR-2248, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1989.
- Yuan S.M. and Ashok K. Agrawala, "An Efficient Communication Structure for Decentralized Algorithms with Fault Tolerance", CS-TR-2206, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1989.
- Sam H. Noh and Ashok K. Agrawala, "Process Timing in UNIX", CS-TR-2205, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1989.
- Ashok K. Agrawala and Shem-Tov Levi, " Distributed Real-Time Operating Systems", McGraw-Hill, Inc. New York, NY 1989.
- Nehmer J., "A Structuring Framework for Distributed Operating Systems", CS-TR-2079, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, July 1988.
- Shem-Tov Levi, Daniel Mossé, and Ashok K. Agrawala, "Allocation of Real-Time Computations under Fault Tolerance Constraints", CS-TR-2018, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, May 1988.
- Shem-Tov Levi, Ashok K. Agrawala, and Satish K. Tripathi, "Introducing the MARUTI Hard Real-Time Operating System", CS-TR-2010, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1988.
- J. Nehmer, "An Object Architecture for Hard Real-Time Systems", CS-TR-2003, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, March 1988.
- Chintamaneni P., Xiaoping George Yuan, Satish K. Tripathi , and Ashok K. Agrawala, "Scheduling Tasks in a Real-Time System", CS-TR-1991, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, February 1988.
- Shem-Tov Levi, Daniel Mossé, and Ashok K. Agrawala, "Resource Allocation under Fault-Tolerance Constraints", Proceedings of the IEEE Real-Time Systems Symposium, Huntsville, AL, 1988
- Shem-Tov Levi, "A Methodology for Designing Distributed, Fault-Tolerant, Reactive, Real-Time Operating Systems", PhD. Dissertation University of Maryland, College Park, MD. 1988.
- Xiaoping George Yuan, Satish K. Tripathi, and Ashok K. Agrawala, "Scheduling in Real-Time Distributed Systems-A Review", CS-TR-1955, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, December 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Temporal Relations and Structures in Real-Time Operating Systems", CS-TR-1954, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, December 1987.

- Shieh Y., Satish K. Tripathi, Chintamaneni P. and Pankaj Jalote, "On Fault Tolerance in Manufacturing Systems",, CS-TR-1939, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, October 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Objects Architecture: A Comprehensive Design Approach for Real-Time, Distributed, Fault-Tolerant, Reactive Operating Systems", CS-TR-1915, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, September, 1987.
- Finkel D. and Satish K. Tripathi, "An Analysis of a Buddy System for Fault Tolerance", CS-TR-1924, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, August 1987.
- Shem-Tov Levi and Ashok K. Agrawala, " On Real-Time Systems Using Local Area Networks", CS-TR-1892, Technical Report, Department of Computer Science, UMIACS-TR-87-35, Technical Report, Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland, July 1987.
- Ashok K. Agrawala and Shem-Tov Levi, "Objects Architecture for Real-Time, Distributed, Fault Tolerant Operating Systems", IEEE Workshop on Real-Time Operating Systems, Cambridge, MA, July 1987, pp. 142-148.
- Shem-Tov Levi and Ashok K. Agrawala, " On Real-Time Operating Systems", CS-TR-1838, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1987.
- Shem-Tov Levi and Ashok K. Agrawala, "Real-Time Programs: Design Implementation and Validation-A Survey", CS-TR-1837, Technical Report, Department of Computer Science, University of Maryland, College Park, Maryland, April 1987.